Performance Monitoring Hardware Will Always Be A Low-Priority, Second-Class Feature Until…

Brinkley Sprunt
Bucknell University
Definition:

EMON:

- Event MONitoring Hardware
- Performance Monitoring Hardware
Outline

- EMON Problems
- The Principal Cause
- An Opportunity and a “Solution” Approach
Event Definition vs. Implementation

- The architect’s event definition is not fully understood by the designer.

- Result: Events that are too “broken” to be useful.

- Example: DTLB Misses on the P6:
  - Architect: Count memory references that miss the DTLB.
  - Designer: Count # times DTLB is referenced, with no match.

  - Problem:
    - Cancelled, conditional uops for string instructions all miss the DTLB.
    - All DTLB miss counts can be unpredictably too high.
 Desired Features vs. Design Constraints

Goal:
- Provide a comprehensive set of events and counters that enable OS and application performance tuning.

Reality:
- Only a very small % of processors will run apps that require EMON.
- It’s very difficult to defend the ROI for EMON hardware.

Directive:
- Define and implement EMON, but you have zero silicon area!

Defense:
- EMON hardware is the key to improving performance post-silicon.

Result:
- EMON is low priority & implemented in the “nooks and crannies”.
Processor Validation

Processor Validation Priorities:

#1 Functional Correctness.
#2 Functional Correctness.
#3 Functional Correctness.
#4 Performance must meet expectations.
...
#N. EMON events must be correct.

Often:
- Too little pre-silicon EMON validation is done.
- Post-silicon EMON validation is thin and done quickly.
- Many events remain unvalidated and undocumented.
- Documentation is cryptic, partial, and sometimes wrong.
The Principal Cause

Processor Design Priorities:

#1 Meet the functional and performance expectations of the market.

#2 Provide compelling features to attract customers, e.g.:
   - SIMD
   - SMP, SMT, & CMP
   - 64-bit support
   - Improved virtual machine support

EMON Return On Investment:

- EMON ROI is vanishingly small.
- No mainstream user of EMON hardware.
An Opportunity and a “Solution” Approach

Opportunity:
- Mainstream (mass-market) SMP, SMT, CMP systems.
- In these systems:
  - Tasks **concurrently share processor resources**.
  - Contrast with uni-processor, non-threaded systems where a task is allocated the whole processor.

Performance can be significantly improved by using dynamic task performance data to guide task scheduling:
- Which tasks should concurrently share the same physical processor in an SMT system?
- Which tasks should concurrently execute on different cores within the same package in a CMP system?
“Symbiotic” Task Scheduling

- Monitor task performance and either:
  - Use task performance characteristics to categorize and schedule tasks together that “like” each other.
  - Measure performance of random, fair task schedules and pick highest throughput schedules for longer-term execution.

- Symbiotic scheduling was initially investigated by the Simultaneous Multithreading Project at University of Washington.

- Symbiotic scheduling is the “killer app” that will bring EMON hardware into the mainstream.

*We should foster the development of operating systems that dynamically tune task scheduling using real-time processor performance measurements.*